

AN EVALUATION OF A MODIFIED BINARY SEARCH
PROCEDURE FOR USE WITH THE BRUCETON METHOD
IN SENSITIVITY TESTING

by

Donald Lee Hicks

United States Naval Postgraduate School



THESIS

AN EVALUATION OF A MODIFIED BINARY SEARCH PROCEDURE
FOR USE WITH THE BRUCETON METHOD
IN SENSITIVITY TESTING

by

Donald Lee Hicks

September 1970

This document has been approved for public
release and sale; its distribution is unlimited.

T135604

An Evaluation of a Modified Binary Search Procedure
for use with the Bruceton Method
in Sensitivity Testing

by

Donald Lee Hicks
Major, United States Marine Corps
B.S., United States Naval Academy, 1957

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1970

Heaven H. 2601
C. 1

ABSTRACT

Two methods of obtaining sensitivity data were simulated on an electronic computer for the purpose of comparing the accuracy of the estimates of the parameters of an underlying cumulative normal response function. The first method simulated the standard Bruceton procedure while the second used a modified binary search routine with a portion of the sample in order to obtain maximum likelihood estimates of the input parameters for use in a follow-on Bruceton test.

The results showed both methods to be effective in estimating the mean but with slightly more variability in the estimates obtained by the second procedure. Both methods underestimated the standard deviation - again with more variability in the estimates obtained by the second procedure. When the prior parameter estimates were unknown and the applicable stimulus level bounded, the second method yielded estimates favorably comparable to those expected from the Bruceton procedure with suitable prior input estimates.

TABLE OF CONTENTS

I.	INTRODUCTION	7
II.	THE MODEL	9
III.	TESTING METHODS	11
	A. BRUCETON METHOD	11
	1. Description	11
	2. Discussion	12
	B. BRUCETON METHOD PRECEDED BY SEARCH	13
	1. Description	13
	2. Discussion	19
IV.	SIMULATION	21
	A. DESCRIPTION	21
	B. MEASURES OF EFFECTIVENESS	22
	C. DISCUSSION	23
	D. RESULTS	24
V.	CONCLUSIONS AND RECOMMENDATIONS	26
	A. CONCLUSIONS	26
	1. Estimation of the Mean	26
	2. Estimation of the Standard Deviation	26
	3. Extension of the Search Phase for the Starting Sequence	26
	4. Use of Search Technique	26
	B. RECOMMENDATIONS	26
	1. Reduction of Sample Size	26
	2. Random Selection of Response Function Parameters	27
	COMPUTER PROGRAM	28

BIBLIOGRAPHY 40

INITIAL DISTRIBUTION LIST 41

FORM DD 1473 43

LIST OF FIGURES AND TABLES

Figure		Page
1	Starting Sequence	15
2	S_U Sequence	16
3	S^* Sequence	17
4	S_L Sequence	18
5	Response Sequences and Parameter Estimates	20

Table		Page
I	Table of Experimental Results	25

I. INTRODUCTION

Frequently a statistician is faced with the problem of determining the level of a stimulus which critically affects the performance of a device. The nature of the testing to be discussed is such that once some positive level of the stimulus is applied to the device either a response or a non-response can be immediately observed and, in either case, the device is altered so that a bonafide result cannot be obtained from a second test. Tests of this type are known as sensitivity tests.

One of the many problems besetting those involved in explosives research is that of providing measures and specifying rules to provide for the safe handling and transportation of explosives. Many different types of sensitivity testing apparatus have been developed for laboratory use, the most common being those that subject some quantity of explosive to the impact load of a falling drop-weight from some controllable height. At least as late as October 1965 there remained two important physical problems to be solved; namely, that of establishing a measure of stimulus not highly apparatus-dependent and then that of translation of these results to safe handling rules [1]. These problems are not addressed in this paper but should be kept in mind when considering the overall problem.

In the early 1940's, a technique for obtaining sensitivity data was developed and used in explosives research at the Explosives Research Laboratory, Bruceton, Pennsylvania which has come to be called synonymously, the Bruceton, Staircase, or "Up and Down" Method.

The aim of this method of testing is to increase the accuracy with which certain critical values of the stimulus may be estimated, notably the median (or mean) and standard deviation. The accuracy of the method

depends in part on the stimulus level at which the first item is tested and the interval spacing for subsequent levels of testing [2].

When the stimulus levels mentioned above cannot be determined prior to testing or when little confidence is placed on the available estimates, a preliminary (or search) phase of testing may be desirable to obtain maximum likelihood estimates prior to employing the Bruceton Method with the remainder of the sample. A procedure to do this is offered as an alternative method.

The comparative accuracies of the two techniques were examined through the use of simulation conducted on a high-speed electronic computer. All parameters and estimates considered as inputs to the simulation were kept within ranges for which the Bruceton Method is considered to yield accurate results [2].

II. THE MODEL

Let x be an applied stimulus level ($x \in [0, \infty)$) and $y = y(x)$ be the associated response ($y \in \{0, 1\}$ where "0" denotes no response and "1" denotes response). At any given stimulus level consider y to be the realization of a Bernoulli random variable, Y , with response probability

$$p(x) = \text{Prob } (Y = 1 | x)$$

The function $p(x)$ is called the response function and is further specified as

$$p(x) \approx 0 \quad x \in [0, a]$$

$$0 < p(x) < 1 \quad x \in (a, b)$$

and
$$p(x) \approx 1 \quad x \in [b, \infty)$$

The intervals $[0, a]$, (a, b) and $[b, \infty)$ are called the zero-response region, the mixed-response region, and the one-response region respectively. It is assumed that $p(x)$ is a monotonely increasing function for stimulus values in the mixed-response region. Thus, $p(x)$ can be considered as the cumulative distribution function for a random variable X such that

$$p(x) = \text{Prob } (X \leq x). \quad [3]$$

In this context the random variable X can be interpreted as a threshold stimulus level, thus

$$\text{Prob } (Y = 1 | x) = \text{Prob } (X \leq x) = p(x)$$

and
$$\text{Prob } (Y = 0 | x) = \text{Prob } (X > x) = 1 - p(x). \quad [3]$$

It is assumed the X is distributed Normal (μ, σ^2) ; that is

$$p(x) = \Phi(x | \mu, \sigma^2)$$

where $\Phi(x | \mu, \sigma^2)$ represents the cumulative normal distribution with mean

μ and variance σ^2 . In particular

$$\text{Prob } (x \leq \mu) = p(\mu) = 0.5. \quad [3]$$

III. TESTING METHODS

A. BRUCETON METHOD

1. Description

Based on intuition or past experiments, the experimenter selects a priori estimates of μ and σ . Call these estimates μ_I and σ_I and let $d = \sigma_I$.

The experimenter tests the first item at or near μ_I . If there is a response the second item is tested at a level d units below μ_I , otherwise the second item is tested at a level d units above μ_I . In the same manner, each of the remaining items is tested at a level d units above or below the previous test level according as there was not or there was a response observed for the previous test. Thus the sample is concentrated about the mean and one would expect nearly equal numbers of responses and non-responses. In fact, the number of non-responses at any level will not differ by more than one from the number of responses at the next higher level [2].

Let N denote the total number of observations of the less frequent event and $n_0, n_1, n_2, \dots, n_k$ denote the frequencies of this event at each level where n_0 corresponds to the lowest level and n_k the highest level at which the less frequent event occurs.

The final estimates of μ and σ are based on the first two moments of the stimulus levels. Since the intervals are equally spaced, these moments can be computed in terms of the sums

$$A = \sum_i i n_i$$

and

$$B = \sum_i i^2 n_i.$$

Let $\hat{\mu}$ be the estimate of μ by this method. Then

$$\hat{\mu} = x' + d \left(\frac{A}{N} \pm \frac{1}{2} \right)$$

where x' represents the lowest level at which the less frequent event occurs [2]. The plus sign is used when the analysis is based on non-responses, and the minus sign when it is based on responses [2].

If $(NB-A^2)/N^2 > .3$ the sample standard deviation is

$$s = 1.620 d \left(\frac{NB-A^2}{N^2} + .029 \right)$$

Otherwise, a more elaborate calculation must be employed and is described in Ref. 2.

To obtain confidence intervals, estimates of the standard deviations of the sample mean and sample standard deviation, say s_m and s_s respectively, are given by

$$s_m = \frac{Gs}{\sqrt{N}}$$

and

$$s_s = \frac{Hs}{\sqrt{N}}$$

where the factors G and H are dependent on the ratio $\frac{d}{s}$ and the position of the mean relative to the testing levels. Plots of these factors are available in Ref. 1.

2. Discussion

Only rarely is the threshold stimulus Z normally distributed. It is usually the case that some scale transformation of Z , say X , is made so that X is normally distributed in the vicinity of the mean. This transformation is done prior to testing to determine μ_I and σ_I . Only after all analysis is completed are the values scaled back to the original stimulus measure [2].

The size of the sample is critical to the accuracy of the estimation. Note that at most only half of the sample is used in the analysis so that, for example, if thirty items are tested the maximum possible value of N is fifteen. The analysis is based on large sample theory which in the case mentioned would be applied to a sample of size fifteen [2] [4].

Unless normality of the variate is assured this method does not yield accurate results for the small and large percentage points. This is unfortunate since in most applications one would be more interested in a small percentage point as a measure of safety and a large percentage point as a measure of reliability. At any rate, an estimate of a percentage point j is

$$j = \hat{\mu} + ks$$

where k is chosen from tables of the standard normal deviate to give the desired percentage [2]. One could then conduct tests in the vicinity of this value to refine the estimate.

B. BRUCETON METHOD PRECEDED BY SEARCH

1. Description

In the event that a priori estimates of μ and σ are not available some economic method of attaining these estimates is desired. A method proposed and described below is a modified binary search technique.

Again, the assumption is that the threshold stimulus (or some transformation of it) is normally distributed and $p(x)$ can be represented by a cumulative normal distribution.

As noted from the model

$$\text{Prob } (Y = 0 | x \leq a) \approx 1$$

and

$$\text{Prob } (Y = 1 | x \geq b) \approx 1.$$

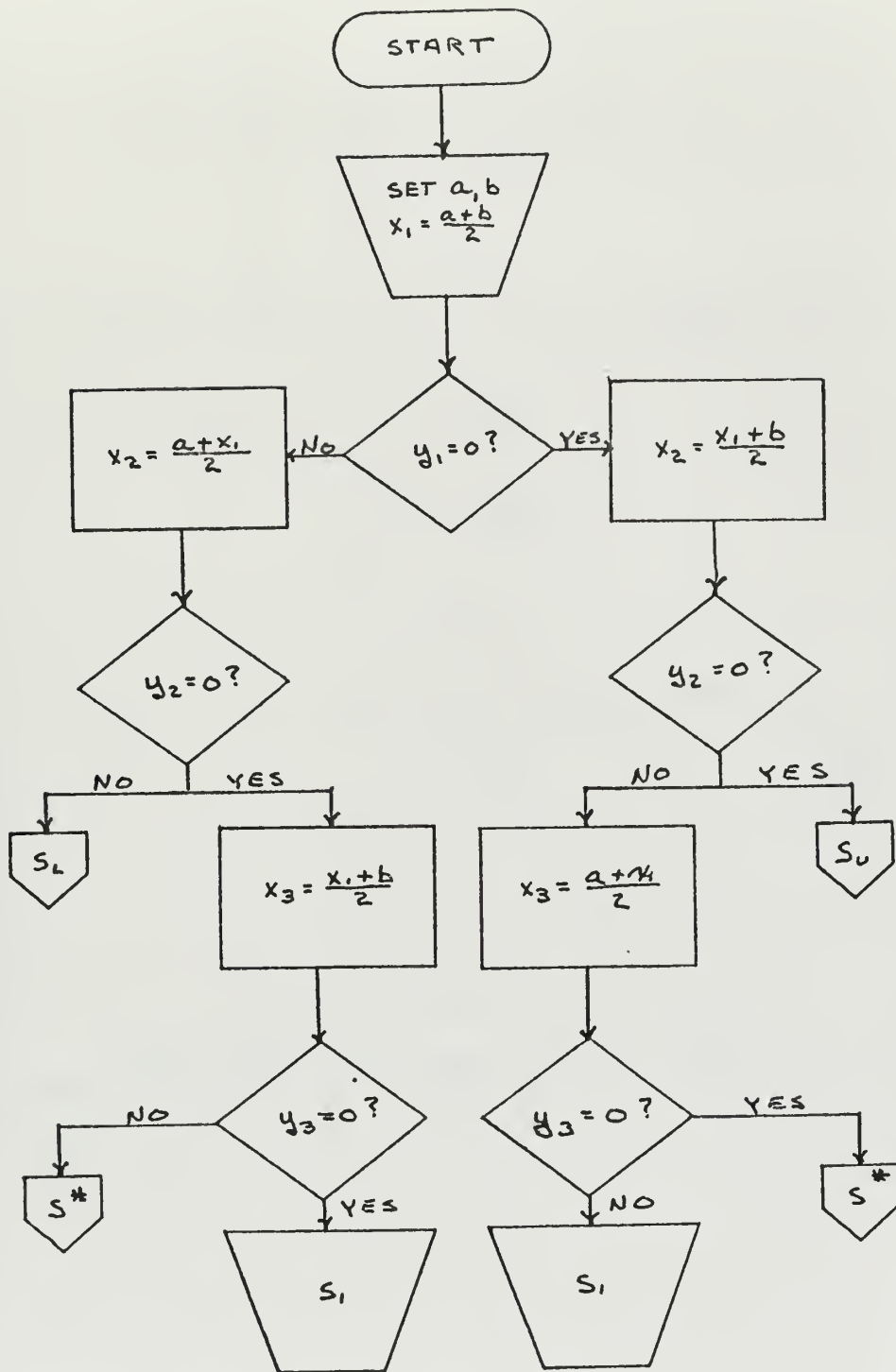
The first step in the procedure, then, is to select values for a and b . (In the case of complete uncertainty these could be the limiting values of the testing apparatus) and commence the binary search starting at

$$x = (a + b)/2.$$

If $p(x)$ were a step function, repetition of this method would locate the step in an interval of any desired length. In general, however, the mixed-response region has non-zero width and a non-response would merely indicate that the applied stimulus is in the mixed response region or below while a response would indicate that it was in the mixed response region or above.

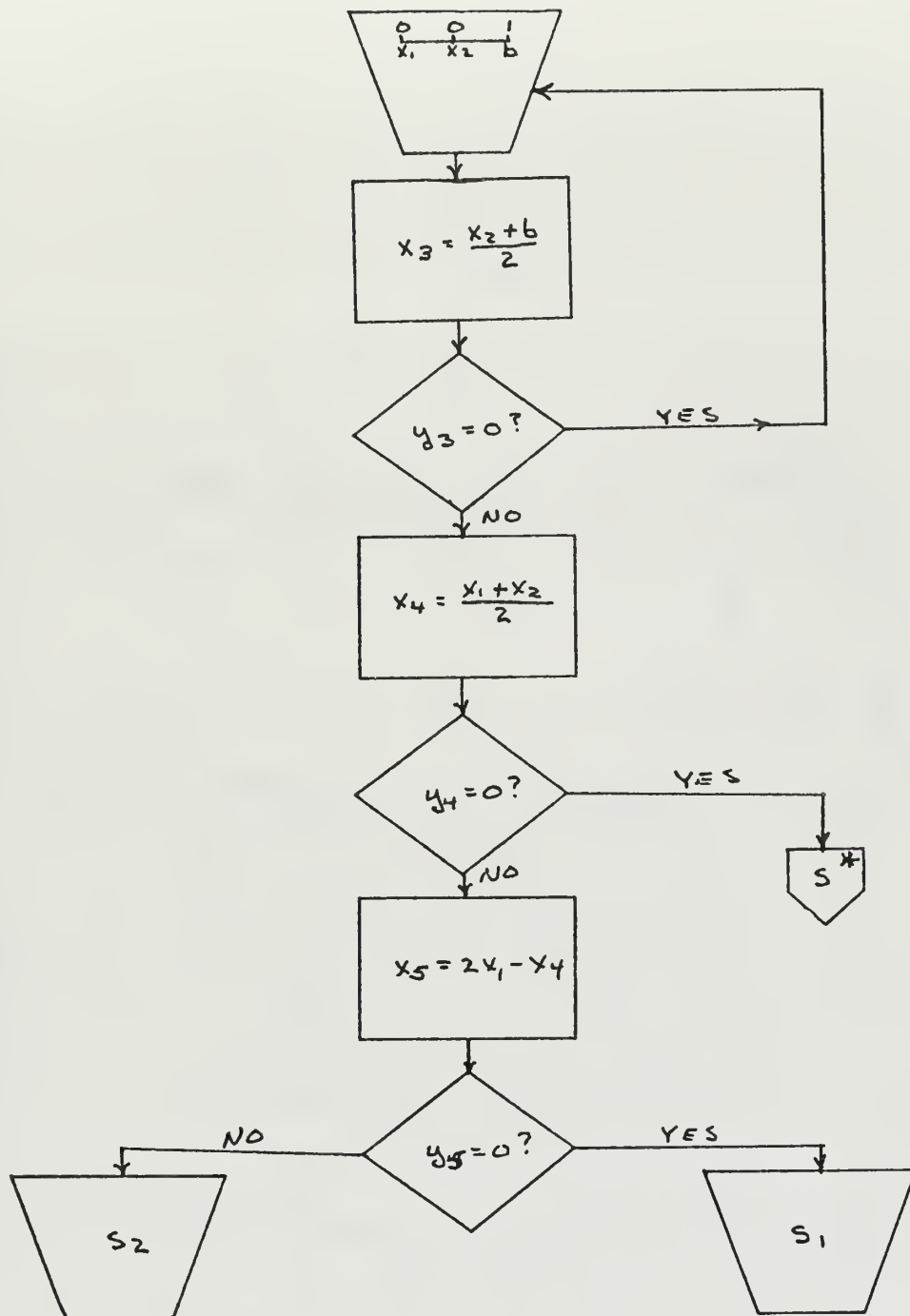
If a test at x_1 yields a response and a test at x_2 yields a non-response while $x_1 < x_2$ it is certain that both x_1 and x_2 are in the mixed response region. This condition is called a response inversion and is the basic indicator for the modified binary search technique. The description of the procedure is best followed by referring to Figures 1 through 4.

Sequence S^* is a cyclic one indicating that a reduction in step size should be taken. Test levels are selected attempting to reproduce this sequence. Failure to do this results in the basic inversion sequence S_0 . Tests are then made at the end of this sequence to result in one of three terminal situations S_1 , S_2 , or S_3 . In the event the mixed response region is relatively narrow and near a or b , several binary reductions may be necessary to reproduce S^* or one of the terminal situations. These circumstances are represented by S_L and S_U [3].



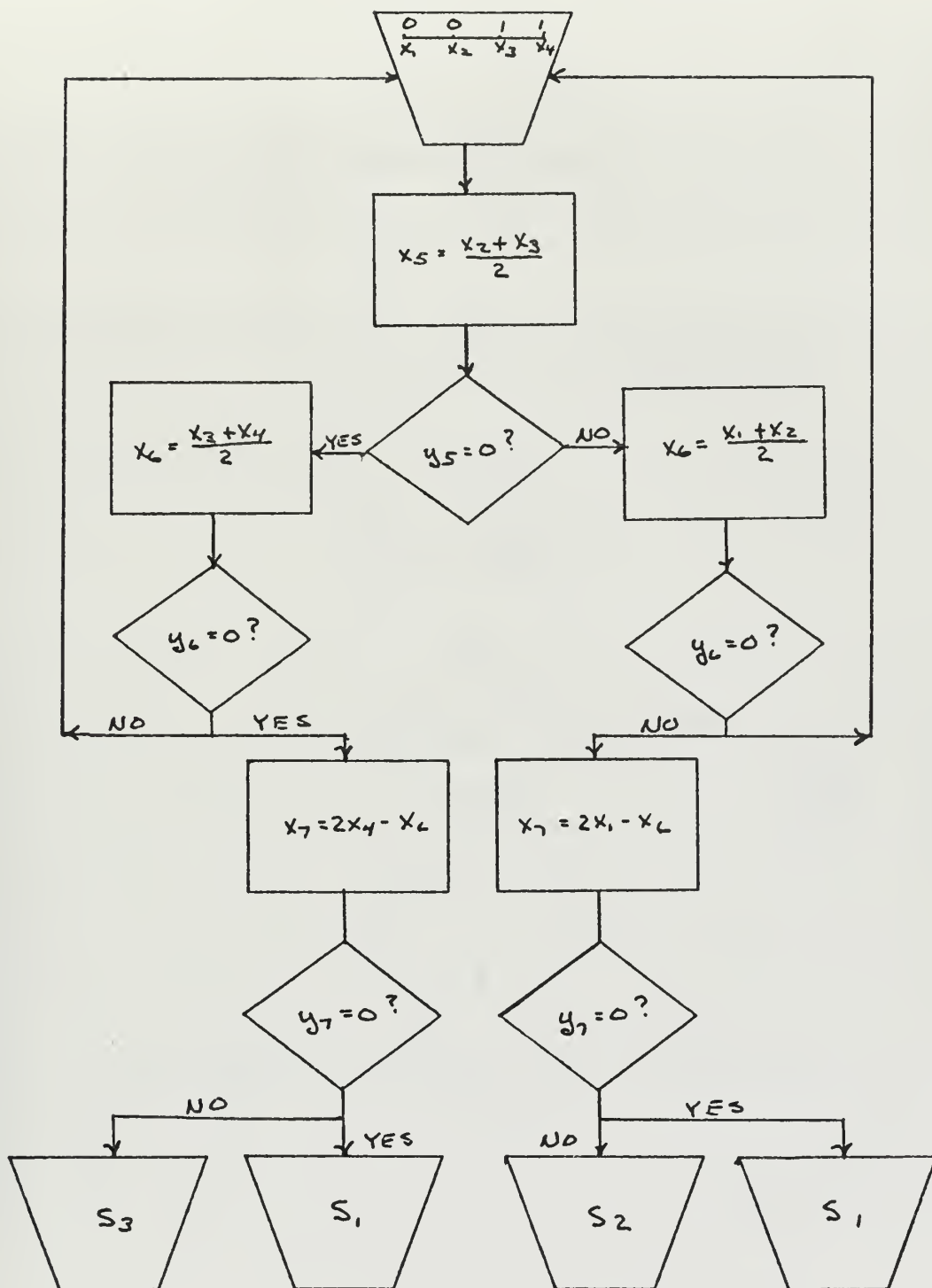
STARTING SEQUENCE

Figure 1



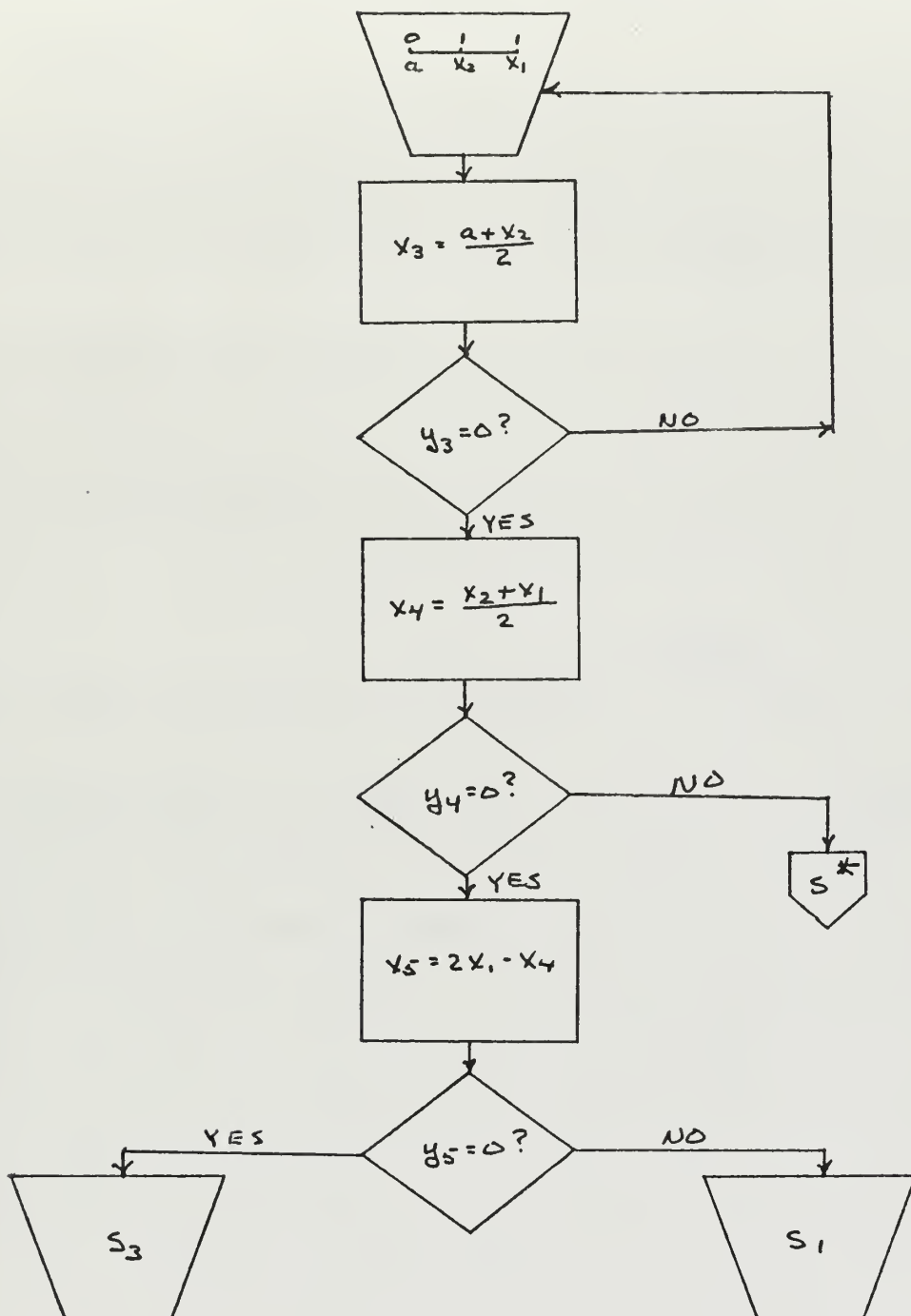
S_U SEQUENCE

Figure 2



S* SEQUENCE

Figure 3



S_L SEQUENCE

Figure 4

Maximum likelihood estimates of μ and σ are available for sequences S_1 , S_2 , and S_3 and developed as described below [3].

2. Discussion

It is assumed that all trials are independent. Thus the probability of the sequence S_1 is

$$\begin{aligned} \text{Prob}(S_1) &= \text{Prob}(Y_1=0, Y_2=0, Y_3=1, Y_4=0, Y_5=1, Y_6=1 | X_1, X_2, X_3, X_4, X_5, X_6) \\ &= \prod_{i=1}^6 \text{Prob}(Y_i = y_i | x_i) \end{aligned}$$

where

$$\begin{aligned} \text{Prob}(Y_i = y_i | x_i) &= \varphi(x_i) \text{ if } y_i = 1 \\ &= 1 - \varphi(x_i) \text{ if } y_i = 0 \end{aligned}$$

and

$$\varphi(x_i) = \text{Prob}(X_i \leq x_i) = \int_{-\infty}^{x_i} \frac{1}{\sqrt{2\pi} \sigma} e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}} dx.$$

Maximum likelihood estimates for μ and σ can then be established using standard normal tables for each of the terminal situations. These estimates are indicated on Figure 5.

S^*



S_0



$$\mu_s \approx x_2 + \frac{\Delta x}{2}$$

$$\sigma_s \approx 1.6 \Delta x$$

S_1



$$\mu_s \approx x_2 + \frac{\Delta x}{2}$$

$$\sigma_s \approx 1.3 \Delta x$$

S_2



$$\mu_s \approx x_5 - \frac{\Delta x}{4}$$

$$\sigma_s \approx 6 \Delta x$$

S_3



$$\mu_s \approx x_6 + \frac{\Delta x}{4}$$

$$\sigma_s \approx 6 \Delta x$$

RESPONSE SEQUENCES AND PARAMETER ESTIMATES

Figure 5

IV. SIMULATION

A. DESCRIPTION

All simulated experiments were conducted on an IBM 360/67 computer using the FORTRAN IV programming language. The basic program is attached. The response function $p(x)$ used was cumulative normal with $\mu = 30$ and $\sigma = 3$.

The sample size was kept at seventy for each experiment to provide some assurance that the analytical sample would be suitable for large sample analysis.

The basic test procedure was to draw a random number on the unit interval and compare this to $F(x)$, a function of a standard normal variate specified as

$$F(x) = \frac{1}{2} \left[1 - \operatorname{erf} \left(\frac{x}{\sqrt{2}} \right) \right] \quad \text{if } x < 0$$

and

$$F(x) = \frac{1}{2} \left[1 + \operatorname{erf} \left(\frac{x}{\sqrt{2}} \right) \right] \quad \text{if } x \geq 0$$

where

$$\operatorname{erf}(v) = \frac{2}{\sqrt{\pi}} \int_0^v e^{-t^2} dt.$$

(The function subprogram erf is an IBM-supplied subprogram.) If the random number was less than or equal to $F(x_i)$ then a response was counted for the i^{th} level; otherwise a non-response was counted.

Six different cases were tested using the straight Bruceton procedure (METHOD 1) with two different input estimates of μ and three different input estimates of σ . Case 1 considered exact estimates; i.e., $\mu_I = \mu$ and $\sigma_I = \sigma$. Case 2 considered $\mu_I = \mu - 6$ and $\sigma_I = \sigma$.

Cases 3 and 4 considered $\mu_I = \mu$ and $\sigma_I = \sigma/2, 2\sigma$ respectively while Cases 5 and 6 repeated Cases 3 and 4 except $\mu_I = \mu - 6$. For each of the six cases 1000 experiments were conducted each utilizing a different sequence of random numbers.

The search procedure (METHOD 2) was then incorporated into each of the above six cases using the a priori estimates, μ_I and σ_I , to determine estimates for stimulus levels a and b and thereby the size of the binary reduction as indicated in Figure 1. The program then followed the flow shown in Figures 1 through 4 until either a terminal sequence was reached or the search was arbitrarily terminated as discussed in subparagraph C below. The Bruceton procedure was then used until the sample was exhausted.

The final case, Case 7, indicated complete lack of knowledge of μ and σ but considered the upper and lower stimulus level limits of the test apparatus to be 100 and 0 respectively.

B. MEASURES OF EFFECTIVENESS

At the completion of all experiments for each case, several measures were obtained for comparison. First, average values of the parameters were determined to be

$$\bar{\hat{\mu}} = \sum_i \hat{\mu}_i / N$$

and

$$\bar{\hat{\sigma}} = \sum_i \hat{\sigma}_i / N$$

where $\hat{\mu}_i$ and $\hat{\sigma}_i$ are the a posteriori estimates of μ and σ for the i^{th} experiment and N , the number of experiments used. Next, as measures of variability

$$s_{\hat{\mu}}^2 = \sum_i (\hat{\mu}_i - \mu)^2 / n-1$$

and

$$s_{\hat{\sigma}}^2 = \sum_i (\hat{\sigma}_i - \sigma)^2 / n-1$$

were calculated. In addition, the program listed the maximum and minimum estimates of both μ and σ .

C. DISCUSSION

In Chapter III it was noted that sequences S^* , S_U , and S_L are cyclic. In order to simplify the program it was necessary to artificially terminate these situations at some point and calculate the input values for the Bruceton test. The estimate of μ used was

$$\mu_s = (x_1 + x_2)/2$$

where x_1 and x_2 are adjacent testing levels and $x_2 > x_1$ with $y_1 = 0$ and $y_2 = 1$. The estimate of σ used was

$$\sigma_s = (x_2 - x_1)/2$$

for Cases 1 through 6 and

$$\sigma_s = (x_2 - x_1)/6$$

for Case 7. The former estimate of σ was chosen arbitrarily while the latter estimate was based on the estimate of the mixed response region being 6σ . While the number of terminations of this type was insignificant for the first six search cases, in the final case over 600 experiments were thus terminated requiring the program to be expanded to permit more recycling. The point is that the artificial termination does not represent the search procedure. This problem would not arise in field experimentation until either the sample was exhausted or the step size reduction of stimulus level indicated was too narrow to be measured or controlled by the test apparatus.

Also in the interest of program simplification those experiments for which

$$\frac{NF - A^2}{N^2} \leq .3$$

were not used for analyses. This limitation invalidated the measures of effectiveness for the Bruceton cases where $\sigma_I = 2\sigma$.

D. RESULTS

The results of the simulation are listed in Table I. It is questionable that the measures listed under Method 1 are valid for Cases 4 and 6 in that only .381 and .393 of the possible experiments were used. These two cases and Case 4 under Method 2 (where .661 of the possible experiments were used) are the only ones for which $\bar{\hat{\sigma}} > \sigma$.

In general the extreme estimates are more widely separated and the variability of $\hat{\sigma}$ is greater in Method 2.

Estimates of μ range from 27.8823 to 31.7647 for Method 1 and 27.937 to 31.91 for Method 2.

Estimates of σ range from .8741 to 6.5027 for Method 1 and .3498 to 9.8328 for Method 2.

The lowest average $\hat{\mu}$, 29.9113, was obtained under Method 1, Case 5, while the highest average $\hat{\mu}$, 30.1175, was obtained under Method 2, Case 3.

The lowest average $\hat{\sigma}$, 2.3748, was obtained under Method 2, Case 5, while the highest average $\hat{\sigma}$, 2.9474, was obtained under Method 1, Case 5. (Case 6 is not counted under Method 1 nor is Case 4 under both methods.)

TABLE OF EXPERIMENTAL RESULTS

		METHOD 1		METHOD 2	
		$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$
CASE 1					
$\mu_I = 30$	AVE	30.0067	2.8320	30.0117	2.8609
$\sigma_I = 3$	MAX	31.7647	5.7904	31.7813	5.9343
$a = 18$	MIN	28.5000	1.6089	28.2187	1.1241
$b = 42$	VAR	.2523	.4128	.2514	.5831
CASE 2					
$\mu_I = 24$	AVE	29.9641	2.9040	30.0317	2.8819
$\sigma_I = 3$	MAX	31.6765	5.8249	31.6875	9.1369
$a = 12$	MIN	28.3235	1.6250	28.1976	.9512
$b = 36$	VAR	.2656	.4225	.2666	.6336
CASE 3					
$\mu_I = 30$	AVE	30.0295	2.7216	30.1175	2.8615
$\sigma_I = 1.5$	MAX	31.6071	6.1197	31.9100	7.7997
$a = 24$	MIN	28.4118	.8741	28.5950	.8697
$b = 36$	VAR	.2046	.7409	.2693	.9081
CASE 4					
$\mu_I = 30$	AVE	29.9683	3.5424	29.9750	3.0721
$\sigma_I = 6$	MAX	31.4571	6.0266	31.6875	6.3569
$a = 6$	MIN	28.0286	--	27.9370	1.6170
$b = 54$	VAR	.2574	.4639	.2619	.4522
CASE 5					
$\mu_I = 24$	AVE	29.9113	2.9474	29.9363	2.3748
$\sigma_I = 1.5$	MAX	31.4773	5.9257	31.4063	7.9507
$a = 18$	MIN	28.2353	.9452	28.1961	.3498
$b = 30$	VAR	.2220	.8748	.2184	1.4889
CASE 6					
$\mu_I = 24$	AVE	29.9493	3.5438	30.0247	2.8398
$\sigma = 6$	MAX	31.4118	6.5027	31.5756	6.4201
$a = 0$	MIN	27.8823	--	28.2552	1.1252
$b = 48$	VAR	.2639	.4785	.2490	.6300
CASE 7					
--	AVE	--	--	30.0123	2.7280
--	MAX	--	--	31.8229	9.8328
$a = 0$	MIN	--	--	27.9541	.5082
$b = 100$	VAR	--	--	.2628	2.1041

TABLE I

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

1. Estimation of the Mean

Both methods estimate the mean effectively.

2. Estimation of the Standard Deviation

Both methods tend to under-estimate the standard deviation with no predictable bias and are therefore unsuitable for use in safety or reliability statements. This conclusion agrees with the findings of Hampton [4] as it pertains to the Bruceton Method.

3. Extension of the Search Phase for the Starting Sequence

Termination of the search phase with sequence S_1 in the starting sequence (see Figure 1) may yield estimates of σ greater than twice the actual value. To avoid this it is advisable to extend the search phase as described in Ref. 3.

4. Use of Search Technique

The search procedure should be used in those cases where there is not independent evidence that the estimate of σ is within the range for which the Bruceton Method is recommended (i.e., $\sigma/2 < \sigma_I < 2\sigma$).

B. RECOMMENDATIONS

Further testing of Method 2 is recommended under the circumstances listed below.

1. Reduction of Sample Size

It would be of interest to reduce the sample size to the point where the effective sample is small, say 15, and compare the Bruceton procedure with the search procedure using the entire sample for the search.

2. Random Selection of Response Function Parameters

A more valid test of both methods would be achieved by randomly selecting values of μ and σ over some range and using the on-line computer facility to conduct the simulation.

COMPUTER PROGRAM

THIS PROGRAM SIMULATES SENSITIVITY TESTING BY BOTH THE BRUCETON METHOD (WHEN IANY=0) AND THE BRUCETON METHOD PRECEDED BY THE MODIFIED BINARY SEARCH (WHEN IANY=1). THE UNDERLYING RESPONSE FUNCTION IS CUMULATIVE NORMAL (30,3). THE INPUT ESTIMATES OF THE MEAN AND THE STANDARD DEVIATION ARE CALLED EXMU AND EXSIG RESPECTIVELY.

THE PRINCIPLE VARIABLE NAMES ARE AS FOLLOWS...

AACT IS THE STIMULUS VALUE AT THE UPPER LIMIT OF THE MIXED RESPONSE REGION.
 BACT IS THE STIMULUS VALUE AT THE LOWER LIMIT OF THE MIXED RESPONSE REGION.
 A AND B ARE ESTIMATES OF AACT AND BACT RESPECTIVELY.
 X(J) IS THE STIMULUS LEVEL OF THE JTH. STIMULUS.
 IXO(J) IS THE CUMULATIVE COUNT OF NON-RESPONSES AT X(J).
 IXX(J) IS THE CUMULATIVE COUNT OF RESPONSES AT X(J).
 IS IS THE SAMPLE SIZE.
 NU IS THE ENTRY NUMBER FOR THE RANDOM NUMBER GENERATOR, UNIF.
 N COUNTS THE NUMBER OF EXPERIMENTS.
 RN IS THE RANDOM NUMBER ON (0,1) RETURNED BY UNIF.
 FOFX IS THE VALUE OF THE RESPONSE FUNCTION RETURNED BY SUBPROGRAMS XNCDF AND SNCDF.
 ISUMO IS THE TOTAL NUMBER OF NON-RESPONSES FOR ONE EXPERIMENT.
 ISUMX IS THE TOTAL NUMBER OF RESPONSES FOR ONE EXPERIMENT.
 NT IS THE MINIMUM OF ISUMO AND ISUMX.
 NS(J) IS THE FREQUENCY OF THE LESS FREQUENT EVENT AT X(J).
 NG(J) REARRANGES NS(J) SO THAT NG(1)=NS(I) WHERE X(I) IS THE LOWEST STIMULUS LEVEL AT WHICH THE LESS FREQUENT EVENT OCCURS.
 AR(J) IS USED TO CALCULATE THE FIRST MOMENT, SUMAP.
 BR(J) IS USED TO CALCULATE THE SECOND MOMENT, SUMBR.
 YPRIME IS THE LOWEST LEVEL AT WHICH THE LESS FREQUENT EVENT OCCURS.
 XMUEST IS THE FINAL ESTIMATE OF THE TRUE MEAN, XMU.
 DEVEST IS THE FINAL ESTIMATE OF THE TRUE STANDARD DEVIATION, XSIG.
 EMU(J) IS THE DIFFERENCE OF XMUEST AND XMU.
 EDEV(J) IS THE DIFFERENCE OF DEVEST AND XSIG.
 SAMAVM AND SAMAVD ARE THE SAMPLE AVERAGE ERRORS OF XMUEST AND DEVEST RESPECTIVELY.
 SAMSQM AND SAMSQD ARE THE AVERAGE MEAN SQUARE ERRORS OF XMUEST AND DEVEST RESPECTIVELY.
 NOGO IS THE NUMBER OF EXPERIMENTS NOT USED IN THE FINAL ANALYSIS

DIMENSION ARRAYS AND FORMAT

SIMULATE BRUCETON FIRST THEN SEARCH

69 IANY=0
 THING=0.

INITIALIZE INTERNAL AND OUTPUT VARIABLES

SET EXPERIMENT COUNTER, SAMPLE SIZE COUNTER, AND NU.

LCOUNT=1
 NU=12371
 103 N=1
 IF(IANY.EQ.0) NBR=0
 IF(IANY.EQ.1) NBR=1

SET INPUT VARIABLES

XMU=30.
 XSIG=3.
 A=0.
 B=100.
 EXMU=50.


```

EXSIG=12.5
A= EXMU-IQ*EXSIG
B= EXMU+IQ*EXSIG
X1=(A+B)/2.
IQ=4
INC=0

```

PROVIDE BRANCH TO STANDARD BRUCETON

```
IF(NBR.EQ.0) GO TO 33
```

CONDUCT SEARCH

```

CALL UNIF(RN,NU)
FOFX=XNCDF(X1,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9500
X2=(B+X1)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X2,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9250
X3=(B+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X3,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9125
X4=(B+X3)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X4,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9063
X5=(B+X4)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 1313
EXMU=(B+X5)/2.
EXSIG=(X5-X4)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
1313 EXMU=(X5+X4)/2.
EXSIG=(X5-X4)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
9063 X5=(X3+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 1314
X6=(X3+X4)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X6,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 1316
EXMU=(X6+X4)/2.
EXSIG=(X6-X3)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
1316 EXMU=(X6+X3)/2.
EXSIG=(X6-X3)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
1314 X6=2.*X2-X5
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X6,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 1315
XB=X5

```



```

      DELX=X5-X2
      EXMU=XB+DELX/2.
      EXSIG=1.3*DELX
      GO TO 7000
1315  XB=X6
      DELX=X2-X6
      EXMU=XB-DELX/4.
      EXSIG=6*DELX
      GO TO 7000
9125  X4=(X2+X1)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X4,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9094
      X5=(X2+X3)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X5,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9047
      X6=(X3+X3)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X6,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9024
      X7=2.*B-X6
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X7,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9012
      XB=X7
      DELX=XB-B
      EXMU=XB+DELX/4.
      EXSIG=6.*DELX
      GO TO 7000
9012  XB=X3
      DELX=X6-XB
      EXMU=XB+DELX/2.
      EXSIG=1.3*DELX
      GO TO 7000
9024  EXMU=(X3+X5)/2.
      EXSIG=(X3-X5)/2.
      EXSIG=2.*EXSIG
      EXSIG=EXSIG/6.
      GO TO 7000
9047  X6=(X4+X2)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X6,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9011
      EXMU=(X5+X2)/2.
      EXSIG=(X5-X2)/2.
      EXSIG=2.*EXSIG
      EXSIG=EXSIG/6.
      GO TO 7000
9011  X7=2.*X4-X6
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X7,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9010
      XB=X6
      DELX=X2-XB
      EXMU=XB+DELX/2.
      EXSIG=1.2*DELX
      GO TO 7000
9010  XB=X7
      DELX=X4-X7
      EXMU=XB-DELX/4.
      EXSIG=6*DELX
      GO TO 7000
9094  X5=2*X1-X4
      NBR=NBR+1
      CALL UNIF(RN,NU)

```



```

FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9003
XB=X4
DELX=X2-X4
EXMU=XB+DELX/2.
EXSIG=1.3*DELX
GO TO 7000
9003 XB=X5
DELX=X1-X5
EXMU=XB-DELX/4.
EXSIG=6.*DELX
GO TO 7000
9250 X3=(A+X1)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X3,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9375
X4=(X1+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X4,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9004
X5=(B+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9005
X6=2*B-X5
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X6,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9006
XB=X6
DELX=X6-R
EXMU=XB+DELX/4.
EXSIG=6*DELX
GO TO 7000
9006 XB=X2
DELX=X5-XB
EXMU=XB+DELX/2.
EXSIG=1.3*DELX
GO TO 7000
9005 EXMU=(X2+X4)/2.
EXSIG=(X2-X4)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
9004 X5=(X1+X3)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 5555
EXMU=(X1+X4)/2.
EXSIG=(X4-X1)/2.
EXSIG=2.*EXSIG
EXSIG=EXSIG/6.
GO TO 7000
5555 X6=2.*X3-X5
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X6,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 9007
XB=X5
DELX=X1-XB
EXMU=XB+DELX/2.
EXSIG=1.3*DELX
GO TO 7000
9007 XB=X6
DELX=X3-XB
EXMU=XB-DELX/4.
EXSIG=6*DELX
GO TO 7000

```



```

9375  X4=2.*A-X3
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X4,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9376
      XB=X3
      DELX=X1-XB
      EXMU=XB+DELX/2.
      EXSIG=1.3*DELX
      GO TO 7000
9376  XB=X4
      DELX=A-XB
      EXMU=XB-DELX/4.
      EXSIG=6*DELX
      GO TO 7000
9500  X2=(A+X1)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X2,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9501
      X3=(X1+B)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X3,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 5556
      X4=2.*B-X3
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X4,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 5554
      XB=X4
      DELX=X4-XB
      EXMU=XB+DELX/4.
      EXSIG=6*DELX
      GO TO 7000
5554  XB=X1
      DELX=X3-XB
      EXMU=XB+DELX/2.
      EXSIG=1.3*DELX
      GO TO 7000
5556  X4=(X1+X2)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X4,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 5557
      X5=(X1+X3)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X5,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 5553
      X6=2.*X3-X5
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X6,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 5559
      XB=X6
      DELX=XB-X3
      EXMU=XB+DELX/4.
      EXSIG=6*DELX
      GO TO 7000
5559  XB=X1
      DELX=X5-X1
      EXMU=XB+DELX/2.
      EXSIG=1.3*DELX
      GO TO 7000
5553  EXMU=(X4+X1)/2.
      EXSIG=(X1-X4)/2.
      EXSIG=EXSIG/3.
      GO TO 7000
5557  X5=(A+X2)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)

```



```

FOFX=XNCDF(X5,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 5561
X6=(X2+X4)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X6,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 121
X7=(X4+X1)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X7,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 122
X8=2.*X1-X7
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X8,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 123
EXMU=X8+(X8-X1)/4.
EXSIG=6.*(X9-X1)
GO TO 7000
123 EXMU=(X4+X7)/2.
EXSIG=1.3*(X7-X4)
GO TO 7000
122 X8=(X6+X4)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X8,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 124
X9=(X4+X7)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X9,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 125
EXMU=(X4+X9)/2.
EXSIG=1.3*(X9-X4)
GO TO 7000
125 EXMU=(X8+X4)/2.
EXSIG=(X4-X8)/6.
GO TO 7000
124 X9=(X2+X6)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X9,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 126
EXMU=(X6+X8)/2.
EXSIG=(X8-X6)/6.
GO TO 7000
126 EXMU=(X9+X6)/2.
EXSIG=1.3*(X6-X9)
GO TO 7000
121 X7=(X5+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X7,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 127
X8=(X6+X2)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X8,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 128
X9=(X6+X4)/2.
NBR=NBR+1
CALL UNIF(RN,NU)
FOFX=XNCDF(X9,XMU,XSIG)
IF(RN.LE.FOFX) GO TO 129
EXMU=(X6+X9)/2.
EXSIG=1.3*(X9-X6)
GO TO 7000
129 EXMU=(X8+X6)/2.
EXSIG=(X6-X8)/6.
GO TO 7000
128 X9=(X7+X2)/2.

```



```

    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X9,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 130
    EXMU=(X2+X8)/2.
    EXSIG=(X8-X2)/6.
    GO TO 7000
130  EXMU=(X9+X2)/2.
    EXSIG=1.3*(X2-X9)
    GO TO 7000
127  EXMU=(X7+X2)/2.
    EXSIG=1.3*(X2-X7)
    GO TO 7000
5561 X6=2.*A-X5
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X6,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 5560
    XB=X5
    DELX=X5-A
    EXMU=XB+DELX/2.
    EXSIG=1.3*DELX
    GO TO 7000
5560 XB=X5
    DELX=A-X6
    EXMU=XB-DELX/4.
    EXSIG=6*DELX
    GO TO 7000
9501 X3=(A+X2)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X3,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9503
    X4=(X1+X2)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X4,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9504
    X5=2.*X1-X4
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X5,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9080
    XB=X5
    DELX=XB-X1
    EXMU=XB+DELX/4.
    EXSIG=6*DELX
    GO TO 7000
9080 XB=X2
    DELX=X4-X2
    EXMU=XB+DELX/2.
    EXSIG=1.3*DELX
    GO TO 7000
9504 X5=(X3+X2)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X5,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9081
    X6=(X2+X4)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X6,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9082
    X7=2.*X4-X5
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X7,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9083
    XB=X7
    DELX=X7-X4
    EXMU=XB+DELX/4.
    EXSIG=6*DELX

```



```

GO TO 7000
9033 XR=X2
    DELX=X6-X2
    EXMU=XR+DELX/2.
    EXSIG=1.3*DELX
    GO TO 7000
9032 EXMU=(X5+X2)/2.
    EXSIG=(X2-X5)/6.
    GO TO 7000
9031 X6=(X3+A)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X6,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9034
    EXMU=(X3+X5)/2.
    EXSIG=(X5-X3)/6.
    GO TO 7000
9034 X7=2.*A-X6
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X7,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9035
    XB=X6
    DELX=X3-X5
    EXMU=XB+DELX/2.
    EXSIG=1.3*DELX
    GO TO 7000
9035 XB=X7
    DELX=A-X7
    EXMU=XB-DELX/4.
    EXSIG=6*DELX
    GO TO 7000
9503 X4=(X3+A)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X4,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9507
    X5=(X2+X3)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X5,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9509
    X6=2.*X2-X5
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X6,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9510
    XR=X6
    DELX=X6-X2
    EXMU=XR+DELX/4.
    EXSIG=6*DELX
    GO TO 7000
9510 XR=X3
    DELX=X5-X3
    EXMU=XR+DELX/2.
    EXSIG=1.3*DELX
    GO TO 7000
9509 X6=(X3+X4)/2.
    NBR=NBR+1
    CALL UNIF(RN,NU)
    FOFX=XNCDF(X6,XMU,XSIG)
    IF(RN.LE.FOFX) GO TO 9511
    EXMU=(X6+X3)/2.
    EXSIG=(X3-X6)/2.
    EXSIG=2.*EXSIG
    EXSIG=EXSIG/6.
    GO TO 7000
9511 EXMU=(X4+X6)/2.
    EXSIG=(X6-X4)/2.
    EXSIG=2.*EXSIG
    EXSIG=EXSIG/6.
    GO TO 7000

```



```

9507 X5=(X4+A)/2.
      NBR=NBR+1
      CALL UNIF(RN,NU)
      FOFX=XNCDF(X5,XMU,XSIG)
      IF(RN.LE.FOFX) GO TO 9508
      EXMU=(X4+X5)/2.
      EXSIG=(X4-X5)/2.
      EXSIG=2.*EXSIG
      EXSIG=EXSIG/6.
      GO TO 7000
9508 EXMU=(X5+A)/2.
      EXSIG=(X4-X5)/2.
      EXSIG=2.*EXSIG
      EXSIG=EXSIG/6.
      GO TO 7000
7000 XB=0.
      DELX=0.
      WRITE(6,1004) EXMU,EXSIG
      IF(EXSIG.LT.0.) EXSIG=-EXSIG
      OPT=0.
      DIF=0.
      XINC=0.
      OPT=EXMU-4.*EXSIG
      IF(OPT.LT.A) GO TO 1003
      XINC=(OPT-A)/EXSIG
      INC=XINC/1
      DIF=XINC-INC
      IF(DIF.GE..5) INC=INC+1
      IF(DIF.LT..5) INC=INC
      GO TO 1001
1003 A=OPT
      INC=0
1001 M=IQ+INC+1
33   IS=70-NBR
      M=IQ+INC+1

```

CONDUCT BRUCETON TEST

```

CLEAR ARRAYS
DO 10 I=1,200
  X(I)=0.
  IXO(I)=0
  IXX(I)=0
  NS(I)=0
  NG(I)=0
  SUMAR=0.
  SUMBR=0.
  AR(I)=0.
  BR(I)=0.
10   CONTINUE

LOAD X ARRAY
DO 20 J=1,200
  X(J)=A+(J-1)*EXSIG
20   CONTINUE

CONDUCT EXPERIMENT
30   CALL UNIF(RN,NU)
      FOFX=XNCDF(X(M),XMU,XSIG)
      IF(RN.GT.FOFX) GO TO 40
      IXX(M)=IXX(M)+1
      M=M-1
      N=N+1
50   IF(N.GT.IS) GO TO 60
      GO TO 30
40   IXO(M)=IXO(M)+1
      M=M+1
      N=N+1
      GO TO 50

```

PERFORM BRUCETON ANALYSIS

COUNT RESPONSES AND NON-RESPONSES

```

60  ISUMX=0
    ISUMO=0
    DO 14 J=1,200
    ISUMX=ISUMX+IXX(J)
    ISUMO=ISUMO+IXO(J)
    NS(J)=0
    AR(J)=0.
    BR(J)=0.
    NG(J)=0
14  CONTINUE

```

DETERMINE LESS FREQUENT EVENT AND LOAD NS

```

    IF (ISUMX.LE.ISUMO) GO TO 15
    NT=ISUMO
    IFLAG=0
    DO 21 J=1,200
    NS(J)=IXO(J)
21  CONTINUE
    GO TO 16
15  NT=ISUMX
    IFLAG=1
    DO 22 J=1,200
    NS(J)=IXX(J)
22  CONTINUE

```

DETERMINE FIRST AND SECOND MOMENTS

```

16  JCOUNT=1
17  IF (NS(JCOUNT).GT.0) GO TO 18
    JCOUNT=JCOUNT+1
    IF (JCOUNT.GE.200) GO TO 104
    GO TO 17
18  MCOUNT=200-JCOUNT
    DO 19 J=1,MCOUNT
    NG(J)=NS(JCOUNT+J-1)
    AR(J)=(J-1)*NG(J)
    SUMAR=SUMAR+AR(J)
    BR(J)=((J-1)**2)*NG(J)
    SUMPR=SUMPR+BR(J)
19  CONTINUE
    YPRIME=X(JCOUNT)

```

CALCULATE ESTIMATES OF MEAN AND STANDARD DEVIATION

```

    IF (IFLAG.EQ.0) XMUEST=YPRIME+EXSIG*((SUMAR/NT)+(1./2.))
    IF (.NOT. IFLAG.EQ.0) XMUEST=YPRIME+EXSIG*((SUMAR/NT)-(1.
    SIGFAC=((NT*SUMBR)-(SUMAR**2))/(NT**2)
    IF (SIGFAC.GT..3) GO TO 1000
    EMU(LCOUNT)=0.
    EDEV(LCOUNT)=0.
    NOGO=NOGO+1
    GO TO 104
1000 DEVEST=1.62*EXSIG*(SIGFAC+.029)

```

LOAD EMU AND EDEV

```

    EMU(LCOUNT)=XMUEST-XMU
    EDEV(LCOUNT)=DEVEST-XSIG
    ADDMU=ADDMU+EMU(LCOUNT)
    ADDSIG=ADDSIG+EDEV(LCOUNT)
    ADDMUQ=ADDMUQ+EMU(LCOUNT)**2
    ADDSDQ=ADDSQ+EDEV(LCOUNT)**2
    IF (EMU(LCOUNT).LT.0.) GO TO 91
    IF (EMU(LCOUNT).EQ.0.) GO TO 92
    IMUHI=IMUHI+1
    IF (EMU(LCOUNT).GT.HIMU) HIMU=EMU(LCOUNT)
    IF (.NOT. EMU(LCOUNT).GT.HIMU) HIMU=HIMU
    GO TO 93
92  NOMU=NOMU+1
    GO TO 93
91  IMULO=IMULO+1
    IF (EMU(LCOUNT).LT.SMLO) SMLO=EMU(LCOUNT)

```



```

93  IF(.NOT.EMU(LCOUNT).LT.SMLO) SMLO=SMLO
    IF(EDEV(LCOUNT).LT.0.) GO TO 94
    IF(EDEV(LCOUNT).EQ.0.) GO TO 95
    IDEVHI=IDEVHI+1
    IF(EDEV(LCOUNT).GT.DEVHI) DEVHI=EDEV(LCOUNT)
    IF(.NOT.EDEV(LCOUNT).GT.DEVHI) DEVHI=DEVHI
    GO TO 104
95  NODEV=NODEV+1
    GO TO 104
94  IDEVLO=IDEVLO+1
    IF(EDEV(LCOUNT).LT.DEVLO) DEVLO=EDEV(LCOUNT)
    IF(.NOT.EDEV(LCOUNT).LT.DEVLO) DEVLO=DEVLO
104  JCOUNT=0
    SIGFAC=0.
    XMUFST=0.
    DEVEST=0.
    SUMAR=0.
    SUMBR=0.
    LCOUNT=LCOUNT+1

```

HAVE 1000 EXPERIMENTS BEEN CONDUCTED ?

IF(LCOUNT.LT.1001) GO TO 103

IF EXPERIMENTS COMPLETED CALCULATE AND WRITE RESULTS

```

    EXNOGO=NOGO
    SAMAVM=ADDMU/(1000.-EXNOGO)
    SAMAVD=ADDSIG/(1000.-EXNOGO)
    SAMSQM=ADDMUQ/(999.-EXNOGO)
    SAMSQD=ADDSQD/(999.-EXNOGO)
    IF(IANY.EQ.1) GO TO 35
    IANY=IANY+1
    GO TO 69
35  STOP
    END

```

SUBROUTINE UNIF(RN,NU)

SUBROUTINE RETURNS RANDOM NUMBER UNIFORM ON (0,1).

```

    REAL MOD
    MOD= 2**31
    NR=129*NU+1
    RN=NR/MOD
    IF(RN.LT.0.0) RN=-RN
    NU=NR
    RETURN
    END

```

FUNCTION XNCDF(V,XMU,SX)

FUNCTION SUBPROGRAM CALCULATES CUMULATIVE NORMAL.
X IS AN R.V. WITH MEAN,XMU,AND STANDARD DEVIATION,SX.

```

    ARG=(V-XMU)/SX
    XNCDF=SNCDF(ARG)
    RETURN
    END

```

FUNCTION SNCDF(X)


```
FUNCTION SUBPROGRAM CALCULATES STANDARD CUMULATIVE NORMAL.  
  DATA TEST/0.0/  
  IF(TEST.NE.0.0) GO TO 100  
  SR2= SQRT(2.0)  
  TEST=1.  
100  SNCDF=(1.0+ERF(X/SR2))/2.0  
  RETURN  
  END
```


BIBLIOGRAPHY

1. Boyars, C. and Levine, D., "Drop-Weight Sensitivity Testing of Explosives", Pyrodynamics, v. 6, p. 54, 1968.
2. Dixon, W. J., and Mood, A. M., "A Method for Obtaining and Analyzing Sensitivity Data", Journal of the American Statistical Association, v. 43, p. 109-126, 1948.
3. Tysver, J. B., A Binary Search Procedure for Use in Sensitivity Testing, submitted to a technical journal for publication, U. S. N. Postgraduate School, Monterey, California, July 1970.
4. Naval Ordnance Laboratory Technical Report 66-117, Monte Carlo Investigations of Small Sample Bruceton Tests, by L. D. Hampton, p. 3.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Associate Professor J. B. Tysver, Code 55Ty Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	1
4. Major Donald L. Hicks, USMC 20 Mesa Vista Drive Boise, Idaho 83705	1
5. Library, Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940		2a. REPORT SECURITY CLASSIFICATION Unclassified	
		2b. GROUP	
3. REPORT TITLE An Evaluation of a Modified Binary Search Procedure for Use with the Bruceton Method in Sensitivity Testing			
4. DESCRIPTIVE NOTES (Type of report and, inclusive dates) Master's Thesis; September 1970			
5. AUTHOR(S) (First name, middle initial, last name) Donald Lee Hicks			
6. REPORT DATE September 1970		7a. TOTAL NO. OF PAGES 42	7b. NO. OF REFS 4
8a. CONTRACT OR GRANT NO.		9a. ORIGINATOR'S REPORT NUMBER(S)	
b. PROJECT NO.			
c.		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
d.			
10. DISTRIBUTION STATEMENT This document has been approved for public release and sale; its distribution is unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940	

13. ABSTRACT

Two methods of obtaining sensitivity data were simulated on an electronic computer for the purpose of comparing the accuracy of the estimates of the parameters of an underlying cumulative normal response function. The first method simulated the standard Bruceton procedure while the second used a modified binary search routine with a portion of the sample in order to obtain maximum likelihood estimates of the input parameters for use in a follow-on Bruceton test.

The results showed both methods to be effective in estimating the mean but with slightly more variability in the estimates obtained by the second procedure. Both methods underestimated the standard deviation - again with more variability in the estimates obtained by the second procedure. When the prior parameter estimates were unknown and the applicable stimulus level bounded, the second method yielded estimates favorably comparable to those expected from the Bruceton procedure with suitable prior input estimates.

Security Classification						
KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Sensitivity testing						
Bruceton method						
Maximum likelihood estimates						
Computer simulation						

FORM 1473 (BACK)

1 NOV 65

101-807-6821

44

Security Classification

A-31409

Th
H
52
c.

Thesis
H52687
c.1

Hicks

121710

An evaluation of a
modified binary
search procedure for
use with the Bruceton
method in sensitivity
testing.

18 OCT 75
18 OCT 75

23456
23456

Thesis
H52687
c.1

Hicks

121710

An evaluation of a
modified binary
search procedure for
use with the Bruceton
method in sensitivity
testing.

thesH52687

An evaluation of a modified binary search



3 2768 001 01405 3

DUDLEY KNOX LIBRARY